Atmospheric pollution forecasting. Applications of horizontal and vertical vector fields

David Pearson, Mireille Batton-Hubert

Atmospheric pollution forecasting

Applications of horizontal and vertical vector fields

David W. Pearson* — **Mireille Batton-Hubert****

** EURISE*
Jean Monnet University of Saint-Etienne
23 rue du Docteur Paul Michelon
42023 Saint-Etienne, France
david.pearson@univ-st-etienne.fr

*** Centre SITE*
Ecole Nationale Supérieure des Mines de Saint-Etienne
158 Cours Fauriel
42023 Saint-Etienne Cedex 02, France
batton@emse.fr

ABSTRACT. *In this paper we apply methods from differential geometry, called horizontal and vertical vector fields, to the problem of ozone concentration forecasting. These vector fields allow us to find structures in the data and to define a sort of distance between data points. Our aim is to use this distance to define a confidence measure for the forecast.*

RÉSUMÉ. *Dans cet article nous appliquons des méthodes de géométrie différentielle, appelées champs de vecteurs horizontaux et verticaux, au problème de prévisions de concentrations d'ozone. Ces champs de vecteurs nous permettent de trouver des structures de données et de définir une sorte de distance entre des points de données. Notre objectif est d'utiliser cette distance comme une mesure de confiance de la prévision.*

KEYWORDS: *forecasting, atmospheric pollution, horizontal and vertical vector fields, differential geometry.*

MOTS-CLÉS : *prévision, pollution atmosphérique, champs de vecteurs verticaux et horizontaux, géométrie différentielle.*

---

1. Introduction

The problem that we are working on is a fairly well known one. In brief terms, we wish to forecast the level of atmospheric pollution for a particular geographical point. By geographical point we actually mean a data measuring station where atmospheric pollution data are measured as opposed to a "geographical zone", a town for example. The forecast is usually for an horizon of 24 hours, or even more. There are various ways of approaching this problem, in general they fall into two different categories that in control theory terms we would refer to as distributed parameter and lumped parameter models. The distributed parameter approach attempts to model all the physical, chemical and mechanical processes taking place in the atmosphere. This usually leads to a lot of partial differential equations that can require a lot of computer power to solve (Mounier *et al.*, 2001). Apart from the need for computer power, there are other problems associated with this approach (measurements of initial and boundary data, topographical effects, etc.). We have opted for a lumped parameter approach that we believe to be easier to implement and more realistic in its outlook. In other words, we restrict our attention to a particular point source of data and try to model its behaviour. We use the current state of the system (pollution and meteorological data measured at the station) and the forecast meteorological data as inputs to our model and the output is the forecast pollution level.

In order to develop a reliable model, we have experimented with three methods. In the past we have looked at cased based reasoning (Pearson *et al.*, 2002) and neural networks (Pearson *et al.*, 2003; Pearson *et al.*, To appear). The model presented in this paper is the most up-to-date and, in our opinion, the best of our attempts. It is based on a method of fuzzy clustering introduced by Chiu (Chiu, 1994). The clustering method supplies a set of data points, from a model identification data set, that act as cluster centres. These centres play an important role in our work because we want to be able to associate a degree of confidence to each of these centres.

The precise problem that we are concerned with in this paper is that of determining a distance between data points. For a given input to the model, we want to be able to calculate the cluster centre that is the nearest to it with respect to the distance measure that we propose in the paper. In future work we intend to associate the cluster centre with a degree of confidence. If the output of the model agrees with the class associated to the nearest cluster centre and the cluster centre has a high degree of confidence then we argue that we can attach a high degree of confidence to the forecast. To calculate the degree of confidence for a cluster centre we intend to use the existing database. For each forecast we will have the model output and the nearest cluster centre to the particular day. Each cluster centre is associated to a forecast and so we can calculate how many times the model output and cluster centre are in agreement and provide the correct forecast and also how many times they disagree but the cluster centre gives the correct forecast.

What makes our approach different is that, once the model has been identified then we make use of it in a way other than simply an input/output black box. We make the

assumption that in the identification process, the model has in some way "learned" the structure of the data, i.e. that the data are organised in some specific way. We use a method of differential geometry called horizontal and vertical vector fields in order to investigate this structure. This leads us to define the distance between data points not simply as the Euclidian distance but by respecting the data structure as exhibited by the model. Our approach is geometric and is related to, but not the same as Information Geometry as introduced by Amari (Amari *et al.*, 2000). The main source of inspiration for our approach was Ehresmann (Ehresmann, 1950) and other authors looking at connections and fibre bundles.

Basically, our model is a mapping, π, from the input space to a single output value that lies between 1 and 7, these correspond to classes which are introduced later in the paper. For the data pairs chosen by Chiu's algorithm to be the cluster centres (x_k, y_k), where the x_k are the input vectors and the y_k are the outputs, we will have $\pi(x_k) = y_k$ exactly. However, when we present a new input to the model (today's conditions for example), x_0, there will usually be a slight difference between this data point and all the other centres relating to the inputs x_k. Hence, the calculated output $\pi(x_0) = y_0$ will not fall exactly into one of the classes, i.e. $\pi(x_0) = 1.6$ for example instead of $\pi(x_0) = 2$. The problem is then to determine which centre is the closest to x_0 by using the distance measure.

The paper is split into five main sections. First of all we present the clustering model and data. Then we present some notions about fibres and leaves. Following that, the main section is devoted to horizontal trajectories and distance. Some experimental results are then presented before ending with conclusions and perspectives.

In the following, vectors are always indicated by lower case Latin letters, x for example, and are considered as column vectors, row vectors being indicated by the transpose x^T. Vector components are denoted by superscripts, $x = \left[x^1, x^2, \ldots, x^n\right]^T$, and subscripts are used to differentiate between vectors, x_1, x_2, etc. The elements of a matrix, A, will be referred to as a_{ij}. For a mapping $\pi : \mathbb{R}^n \to \mathbb{R}^m$, $\pi(x) = [\pi_1(x), \ldots, \pi_m(x)]^T$ its differential is denoted by π_*, this is just the matrix of partial derivatives $\frac{\partial \pi_i}{\partial x^j}(x)$, the Jacobian matrix, when a coordinate system has been fixed. The Lie bracket of two vector fields $v_1(x), v_2(x)$ is defined by $[v_1(x), v_2(x)] = v_2(x)_* v_1(x) - v_1(x)_* v_2(x)$, when the vector fields are linear $v_k(x) = A_k x$ for matrices A_k then this becomes $[A_1(x), A_2(x)] = (A_2 A_1 - A_1 A_2)x$ (Isidori, 1989; Olver, 1986). The kernel or null space of a matrix A is denoted $\ker A$.

2. Clustering model and data

We will not go into great detail about the clustering method used for our model. It is based on the method introduced by Chiu (Chiu, 1994) and the interested reader who would like to go into more details about the algorithm is invited to read this well written and accessible paper. We will simply present the main points.

As far as the data are concerned, the agency responsible for monitoring the air quality in the Loire department in France (AMPASEL) has supplied us with a comprehensive database for the years 2001, 2002, 2003 and 2004. We have access to meteorological data (temperature, wind speed, wind direction, etc.) and pollution data (ozone, car exhaust gasses, etc.) all acquired on an hourly basis. As is usually the case, we identify our model parameters on a subset of the data and validate the resulting model on a second subset. Due to the fact that 2001, 2002 and 2004 were relatively non-eventful years whereas 2003 was the year known for its heat wave throughout Europe and the associated high levels of ozone, we used 2001, 2002 and 2003 to identify our model and 2004 to validate it.

After consulting the experts at AMPASEL, reading papers written by fellow researchers and carrying out some experiments ourselves we chose the following 8 variables as inputs for our model:

– x^1 maximum concentration of ozone over the past 24 hour period,
– x^2 minimum concentration of ozone over the past 24 hour period,
– x^3 maximum temperature over the past 24 hour period,
– x^4 minimum temperature over the past 24 hour period,
– x^5 maximum forecast temperature for the following day (midnight to midnight),
– x^6 minimum forecast temperature for the following day (midnight to midnight),
– x^7 average forecast wind speed for the following day (midnight to midnight),
– x^8 average forecast wind direction for the following day (midnight to midnight).

The wind direction is actually a class based on compass points, the first class is $[360 - 11.25, 0 + 11.25]$ and continues around the compass in intervals of $22.5°$, making 16 classes in all. For numerical reasons we normalise all the data to lie in the interval $[0, 5]$, this interval was chosen by trial and error and seems to give good overall results.

The output variable, y, is the forecast maximum level of ozone for the following day (midnight to midnight). We don't realistically expect to be able to forecast a precise value for the concentration based on these data, there are simply too many non-measured phenomena that come into play for that. We therefore aim to forecast a class of ozone pollution. In collaboration with the experts at AMPASEL we have defined the following 7 classes (all values are in ppm):

– if concentration > 250 then $y = 7$
– if 230 < concentration $\leq$ 250 then $y = 6$
– if 190 < concentration $\leq$ 230 then $y = 5$
– if 170 < concentration $\leq$ 190 then $y = 4$
– if 150 < concentration $\leq$ 170 then $y = 3$
– if 120 < concentration $\leq$ 150 then $y = 2$
– if concentration $\leq$ 120 then $y = 1$

The constraint imposed on the model is that it has to be applied at 15 hours GMT and the forecast is for the maximum level of ozone for the following 24 hours counted from midnight to midnight.

The model takes the following form:

$$y(x) = \pi(x) = \sum_{k=1}^{c} \frac{w^k \phi_k(x)}{\sum_{i=1}^{c} \phi_i(x)} \qquad [1]$$

where the functions $\phi_k(x)$ are Gaussian

$$\phi_k(x) = e^{-\alpha \|x - c_k\|^2}$$

here the c_k are the cluster centres, there are c of them, w^k are weights corresponding to output values and α is a parameter. The user has to supply the parameter α and then Chiu's algorithm chooses the cluster centres from the data, so the c_k are in fact data points, and calculates the weights w^k. We note that c, the number of centres, can be (and usually is) much smaller than the number of data examples in the set used to identify the model. In the original algorithm of Chiu, once a centre has been chosen then the associated weight is simply the value of the output class. In (Chiu, 1994) he proposes a first order optimisation of the weights by fitting the model to a subset of the model identification set. Depending on how many examples there are in this subset the problem becomes underdetermined, exact or overdetermined. In what follows in this paper we would like the model to provide exact results for centres, in other words if the input vector x_0 corresponds to a centre c_k that belongs to class y_0 then we require $\pi(x_0) = y_0$. This can be achieved for all the centres by applying Chiu's first order optimisation method to the subset of data examples defined precisely by the c centres. Because there are c centres and c weights w^k the resulting set of equations to be solved will be exact.

3. Fibres and leaves

The model presented in the previous section has the form $y = \pi(x)$. Let us assume that for a specific input x_0 we have $\pi(x_0) = y_0$ and we want to calculate the set of x_i that are mapped to the same output as x_0, i.e. such that $\pi(x_i) = y_0$. Given that the model is based on differentiable functions it is reasonable to try and construct this set of points x_i in a continuous way based on the differentiable structure of π. So we want to calculate a trajectory $x(t)$ for some parameter t (which will later on turn out to be a vector), say $t \in [0, 1]$ such that $\pi(x(t)) = y_0, \forall t \in [0, 1]$ and where $x_i = x(t^i)$ for some discrete values $t^i \in [0, 1]$. The mathematical tool that enables us to do this is called a vertical vector field and it is the principal subject of this section. Although

our model is a mapping $\pi : \mathbb{R}^8 \rightarrow \mathbb{R}$, we present vertical fields in a more general setting.

Given a differentiable mapping $\pi : \mathbb{R}^n \rightarrow \mathbb{R}^m$, with $m < n$, a vector field, v, is said to be vertical if the following is satisfied

$$\pi_*(x)v(x) = 0 \qquad [2]$$

If v is a vertical vector field for the mapping π and x_0 is such that $\pi(x_0) = y_0$ then if a solution is found to the following differential equation where $\dot{x} = \frac{dx}{dt}$

$$\begin{aligned} \dot{x} &= v(x) \\ x(0) &= x_0 \end{aligned}$$

then the solution $x(t)$ for $t \geq 0$ will satisfy $\pi(x(t)) = y_0$. In other words the mapping is constant along the trajectories defined by its vertical vector fields. The action of solving such a differential equation with initial point x_0 is called exponentiating the vector field (Isidori, 1989; Olver, 1986) and is written as

$$x(t) = \exp(tv)x_0$$

If there are several vector fields, $\{v_1, \ldots, v_k\}$, then the solutions can be concatenated together. Starting at the initial point x_0 the trajectory moves in the direction of v_1 for time t^1, the point $x(t^1)$ then serves as the initial point for v_2 and the process repeats until the final vector field. Setting $t = \left[t^1, \ldots, t^k\right]^T$ we can write this as

$$x(t) = \exp(t^k v_k) \exp(t^{k-1} v_{k-1}) \cdots \exp(t^1 v_1) x_0$$

If the differential π_* has full rank m for all values of x in some open subset of $\mathbb{R}^n$ then, within this subset, $n-m$ independent vertical vector fields, $\{v_1, \ldots, v_{n-m}\}$ can be found. As illustrated above, solutions of these vertical fields can be concatenated together and, precisely because they are vertical vector fields, these concatenated solutions will satisfy the following condition

$$\pi(x(t)) = \pi(\exp(t^{n-m} v_{n-m}) \exp(t^{n-m-1} v_{n-m-1}) \cdots \exp(t^1 v_1) x_0) = y_0 \qquad [3]$$

We define the following mapping $\phi_{x_0} : O \to \mathbb{R}^n$, where O is an open neighbourhood of the origin in $\mathbb{R}^{n-m}$

$$\phi_{x_0}(t) = \exp(t^{n-m} v_{n-m}) \exp(t^{n-m-1} v_{n-m-1}) \cdots \exp(t^1 v_1) x_0 \qquad [4]$$

Then, for a sufficiently small neighbourhood O, ϕ_{x_0} is a diffeomorphism (a bijective mapping that is also differentiable) from O onto the surface defined by the set of points mapped to y_0 by π. This set of points is referred to as a leaf and to indicate that the leaf is mapped to y_0 by π we write it as L_{y_0} and $\phi : O \to L_{y_0}$. Each leaf corresponds to a connected component of a fibre of π.

As an illustrative example, consider the function $\pi : \mathbb{R}^2 \to \mathbb{R}$ defined as $\pi(x) = (x^1)^2 + (x^2)^2$. It is easily checked that a vertical field for this function is $v(x) = \left[-x^2, x^1\right]^T$. The system $\dot{x} = v(x)$ with $x(0) = x_0$ when integrated gives the standard result

$$x(t) = \begin{bmatrix} x_0^1 \cos(t) - x_0^2 \sin(t) \\ x_0^1 \sin(t) + x_0^2 \cos(t) \end{bmatrix}$$

It can be verified that $\pi(x(t)) = \pi(x_0)$, in this case the vertical vector field is linear and the solution valid for all values of t. Here the leaves (and indeed the fibres) of this mapping are simply circles of radius $r > 0$ defined by $(x^1)^2 + (x^2)^2 = r^2$.

Of course, the mapping π in our case is the clustering model presented in the previous section (1). It is practically impossible to calculate an everywhere valid vertical vector field (2) for such a nonlinear mapping. For this reason, we have developed a method for calculating a linear approximation to v in (2). Originally, our method was developed for neural networks (Pearson, 1996), but it is equally applicable to the fuzzy clustering model. We have changed the method very slightly for the purposes of this work. The original method of second order required an optimisation. But this obviously tends to slow down the calculations and so we have now adopted a first order method with constraints. We will give a few details of this new method.

The model described in the previous section is a mapping $\pi : \mathbb{R}^n \to \mathbb{R}$ with $n = 8$ precisely. We make the assumption that for x in the subset of interest to us $X \subset \mathbb{R}^n$ the mapping is of full rank, i.e. its differential has rank $n - 1$. This being the case there will be $n-1$ independent vertical vector fields $\{v_1, \ldots, v_{n-1}\}$. We will approximate the $n-1$ vertical fields by linear fields by setting $v_k(x) = A_k x$ for $n-1$ matrices A_k, the problem now is to calculate these matrices. No *a priori* structure is imposed on these matrices, they are not assumed to be symmetric for example. We begin by calculating a basis for the kernel of $\pi_*(x_0)$, where x_0 is the initial point in (4). The best way to do this is by applying the singular value decomposition and obtaining an orthogonal basis (Golub *et al.*, 1983) $\{q_1, \ldots, q_{n-1}\}$. Then we need to calculate the A_k's to satisfy $A_k x_0 = q_k$ for $k = 1, \ldots, n-1$. To these constraints

we add the requirement that the vector fields form an involutive and integrable Lie algebra (Isidori, 1989; Olver, 1986), this can be achieved by setting $[A_i, A_j](x_0) = 0$ for $i, j = 1, \ldots, n-1$. To sum up, we need to calculate $n-1$ matrices to satisfy the following conditions:

$$\begin{aligned} A_k x_0 &= q_k \text{ for } k = 1, \ldots, n-1 \\ (A_j A_i - A_i A_j) x_0 &= 0 \text{ for } i, j = 1, \ldots, n-1 \end{aligned}$$

But if the first of these conditions is satisfied then the second one can be simplified because $A_k x_0 = q_k$ implies that $(A_j A_i - A_i A_j) x_0 = A_j q_i - A_i q_j$. Therefore we have

$$\begin{aligned} A_k x_0 &= q_k \text{ for } k = 1, \ldots, n-1 \\ A_j q_i - A_i q_j &= 0 \text{ for } i, j = 1, \ldots, n-1 \end{aligned}$$

By making use of the Kroneker product of two matrices $A \otimes B$ (Wonham, 1979) these conditions can be rewritten as follows

$$\begin{aligned} (1 \otimes x_0^T) a_k &= q_k \text{ for } k = 1, \ldots, n-1 \\ (1 \otimes q_i^T) a_j - (1 \otimes q_j^T) a_i &= 0 \text{ for } i, j = 1, \ldots, n-1 \end{aligned} \qquad [5]$$

where a_k denotes the n^2 dimensional vector obtained by listing the elements of A_k in lexicographical order $a_{11}, a_{12}, \ldots, a_{ij}, \ldots, a_{nn}$. The equations in (5) represent an underdetermined system of linear equations in $(n-1)n^2$ unknowns. We make use of the singular value decomposition once more in order to calculate a minimum norm solution (Golub *et al.*, 1983).

Once the system (5) has been solved then the resulting vertical fields are valid at the point x_0 and by continuity they will also be valid in a neighbourhood of this point. In practice, when we apply the mapping (4) we add the condition that $|\pi(x(t)) - \pi(x_0)| < \mu$ for some parameter $\mu \geq 0$, when this condition is violated then we recalculate (5) at the new point.

4. Horizontal trajectories and distance

Having presented vertical fields in the previous section, we now turn our attention to horizontal fields. Intuitively, the trajectory defined by a vertical field is mapped *via* π to the same point and so a trajectory corresponding to a horizontal field will be mapped via π to different points. This is true, however there is a problem in that once a set of vertical fields has been chosen, the choice of the complementary horizontal

fields is not unique. One way of making this choice unique is related to connections in fibre bundles (Ehresmann, 1950).

Rather than present the theory of fibre bundles and horizontal fields, there is a lot of it, we will present our practical approach to the problem. It is practical by necessity because we need to calculate trajectories by numerical means. In traditional fibre bundle theory the fibres usually have nice topological and geometrical properties, *i.e.* the fibres are spheres or tori etc. In our case, the mapping is nonlinear and we don't have any *a priori* structure for the fibres and so we have to make use of numerical analysis tools such as differential equation solvers and numerical linear algebra. Our approach was inspired by Ehresmann's theory and remains close to his definition of a connection.

We begin with two points x_0 and x_T and a trajectory between these two points $x(t)$ such that $x(0) = x_0$ and $x(T) = x_T$ for some $T > 0$ defined by the differential equation $\dot{x} = f(x)$. The trajectory is mapped via π onto some trajectory $y(t)$ in the output space with $y(0) = y_0$ and $y(T) = y_T$. We begin by calculating (5) at the point x_0 and decompose the tangent space $f(x_0)$ as follows

$$\dot{x}_{|_{t=0}} = f(x_0) = v + h$$

where $v \in \ker \pi_*(x_0)$. Then we calculate a matrix B such that $Bx_0 = v$ and $B = \sum_{k=1}^{n-1} u^k A_k$ where the A_k are the matrices from the Lie algebra and the u^k are scalars.

Following Ehresmann (Ehresmann, 1950), we define a new trajectory via $\tilde{x} = xs^{-1}$ where s is a linear transformation and is a function of t. Now, in (Ehresmann, 1950) it is shown that for $\dot{\tilde{x}}$ to have no vertical component, i.e. to be horizontal, it is sufficient for s to satisfy the matrix differential equation

$$\dot{s} = sB \qquad [6]$$

where B was calculated above. The effect of this is that as the trajectory $x(t)$ passes through each leaf it is moved along the leaf by the transformation s^{-1} until it reaches a point where the tangent vector, $\dot{\tilde{x}}$ is entirely horizontal. Because the trajectory stays on the leaf it will be mapped to the same value by π i.e. $\pi(\tilde{x}) = \pi(x)$. The method is illustrated in figure 1.

Thus, for the example presented above with $\pi(x) = (x^1)^2 + (x^2)^2$ let us choose

$$f(x) = \begin{bmatrix} -x^1 \\ -2x^2 \end{bmatrix}$$

giving

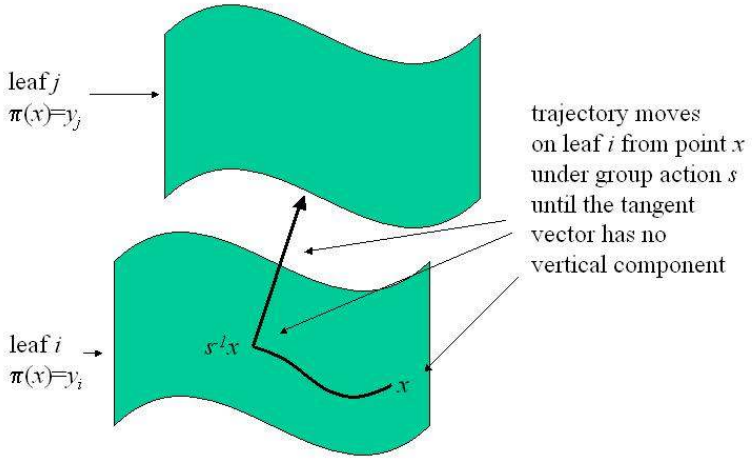

Figure 1. *The notion of a horizontal trajectory*

$$x(t) = \begin{bmatrix} x^1(0)e^{-t} \\ x^2(0)e^{-2t} \end{bmatrix}$$

The vertical field for this example was described above and results in

$$s(t) = \begin{bmatrix} \cos(t) & -\sin(t) \\ \sin(t) & \cos(t) \end{bmatrix}$$

The inverse of this matrix is easily calculated and so we can deduce the horizontal trajectory as follows

$$\tilde{x}(t) = \exp(tf)s^{-1}(t)x(0) = \begin{bmatrix} e^{-t}(x^1(0)\cos(t) + x^2(0)\sin(t)) \\ e^{-2t}(-x^1(0)\sin(t) + x^2(0)\cos(t)) \end{bmatrix}$$

A horizontal trajectory such as that in figure 1 can be seen as a sort of geodesic, in that it covers the shortest distance between two points whilst going straight on with respect to the leaves, i.e. there is no vertical component. When the trajectory gets to the leaf L_{y_T} of course it won't be at the point x_T due exactly to the transformed trajectory. So a vertical trajectory like (4) is applied to go from the point $\tilde{x}(T)$ to x_T

via the shortest path. This is, of course, only valid locally when the two points x_0 and x_T are not too far apart in Euclidian terms. We can, therefore, define the distance between the two points x_0 and x_T as follows

$$d(x_0, x_T) = \int_0^T \|\dot{\tilde{x}}(t)\| dt + \text{distance covered by vertical trajectory} \qquad [7]$$

where $\tilde{x}(0) = x_0$ and $x(T) = x_T$ and the trajectory $\tilde{x}(t)$ is horizontal.

The calculated Lie algebra is only valid in a neighbourhood of the point where it was calculated and so to go from $\tilde{x}(0)$ to $\tilde{x}(T)$ we need to proceed iteratively. We compute the trajectory as a sequence of points as follows

$$x_{k+1} = \exp(\delta t f) x_k$$

and

$$s_{k+1} = s_k \exp(\delta t B_k)$$

with $s_0 = 1$ the identity matrix, B_k calculated as above but based on each new point $\tilde{x}_k$ and $\delta t > 0$ a suitably chosen time increment. The modified (horizontal) trajectory is then given by

$$\tilde{x}_{k+1} = \exp(\delta t f) s_k^{-1} x_k \qquad [8]$$

We choose a very simple form for the vector field f, basically just the straight line segment from the point x_0 to the objective point x_T which is

$$f(x) = x_T - x_0$$

thus $T = 1$ with this choice of field. This makes the trajectory (8) easy to represent because $\exp(\delta t f) x_k = x_k + \delta t (x_T - x_0)$

$$\tilde{x}_{k+1} = s_k^{-1} x_k + \delta t (x_T - x_0)$$

Now we have $\pi(x(t)) = \pi(\tilde{x}(t))$ for $t \in [0, 1]$, but of course $\tilde{x}(T) \neq x_T$ due to the modified trajectory as stated above. But, because $\pi(x(T)) = \pi(\tilde{x}(T))$ then the two points $x(T)$ and $\tilde{x}(T)$ are on the same leaf.

In practice, the difference $|\pi(x(t)) - \pi(\tilde{x}(t))|$ is monitored during the calculation and when this value grows too large the equation for s (6) is reinitialised at $s(0) = 1$ the identity matrix. To calculate the integral in (7) we simply used Simpson's 1/3 rule (Gerald, 1980) which is easy to implement and gives good accurate results for this type of model.

5. Experiments

In this section we present the details of some experiments that we have carried out. There are, of course, many experiments that we could carry out and so we restrict ourselves here to two examples that we find particularly interesting.

Once the model was identified on the 2001, 2002 and 2003 data, we calculated the model results for the 2004 and compared forecast and observed classes. We then chose two days when the model gave erroneous results, one day when the model under forecast the class and the other when it over forecast the class. We thought that it would be particularly interesting if these two days were chronologically close and we found two days that satisfied this criterion:

– 30/07/04 the model forecast class 3 but the day was actually class 4,

– 01/08/04 the model forecast class 5 but the day was actually class 3.

5.1. *Experiment 1 30/07/04*

The input vector for this day was

$$x_0^T = \begin{bmatrix} 153 & 5 & 32.7 & 16 & 33.4 & 16.6 & 1.572 & 0.6875 \end{bmatrix}$$

and the output of the model was 2.572 which resulted in a forecast of class 3.

It would be quite a long calculation to compare the distances between this input vector and each cluster centre using our approach, although it would be feasible to do so of course. We thus decided to limit the calculations by first of all selecting a few cluster centres to be compared. We did this by calculating the ordinary Euclidian distance between the input vector and each cluster centre and ordering the centres by non-decreasing distances. In these terms, the closest cluster centre to the input vector was the following

$$x_T^T = \begin{bmatrix} 140 & 16 & 32 & 15.5 & 33.1 & 17.6 & 1.316 & 0.6875 \end{bmatrix}$$

and corresponds to a day of class 1.

We applied our method to these two vectors. Note that these vectors have been presented in their un-normalised forms so that the reader can get some concrete idea of the type of day involved. The algorithms are of course applied to normalised data, but we de-normalise the results for presentation purposes. However, when distances are subsequently quoted, they refer to the normalised data.

In figure 2 we can see the model output evaluated on the two trajectories x and $\tilde{x}$, they are practically coincident as can be seen. The distance covered by the horizontal trajectory is 0.2041 and the final vertical trajectory on the leaf covers a distance of 0.3345, making a total of 0.5386.

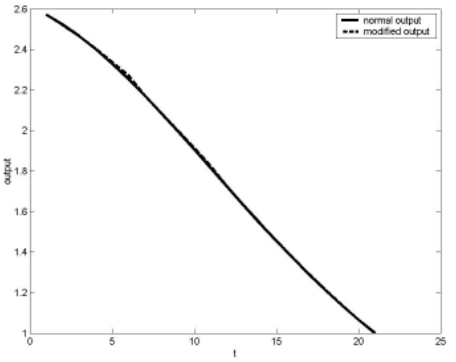

Figure 2. *The model output evaluated on the two trajectories*

We then carried out the same calculations, but using the second nearest cluster centre in terms of Euclidian distance which was the following

$$x_T^T = \begin{bmatrix} 165 & 13 & 31.5 & 17.1 & 33.5 & 17 & 1.192 & 0.6875 \end{bmatrix}$$

and is associated to a day of class 4.

The model output evaluated on the two trajectories x and $\tilde{x}$ is shown in figure 3.

For interest, we present the individual trajectories for the first two components in figures 4 and 5, the other components are similar. In these figures the x^i component is shown as a solid line, the $\tilde{x}^i$ component is the -x- line and the final vertical trajectory on the leaf is the dotted line.

The distance covered by the horizontal trajectory this time is 0.2726 and the vertical field trajectory is 0.1332, giving a total of 0.4058 which is less than than the distance associated with the nearest cluster centre in Euclidian distance terms.

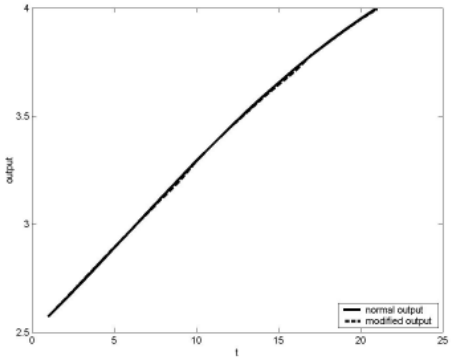

Figure 3. *The model output evaluated on the two trajectories*

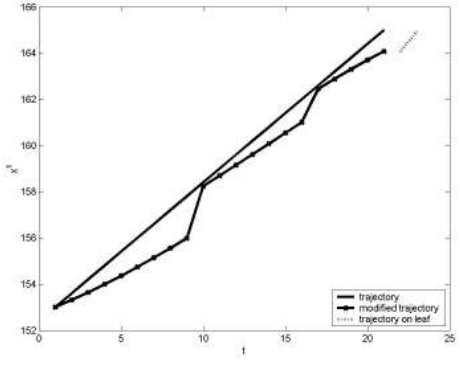

Figure 4. *Individual trajectories*

5.2. *Experiment 2 01/08/04*

The input vector for this day was

$$x_0^T = \begin{bmatrix} 181 & 54 & 34.8 & 18.2 & 35.8 & 19.5 & 1.272 & 0.75 \end{bmatrix}$$

and the nearest cluster centre in Euclidian distance terms was

$$x_T^T = \begin{bmatrix} 181 & 56 & 34.3 & 18.8 & 36.5 & 19.3 & 2.04 & 0.5625 \end{bmatrix}$$

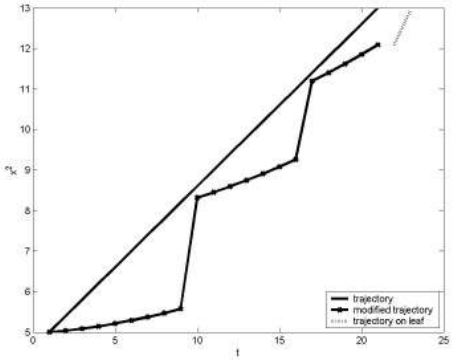

Figure 5. *Individual trajectories*

which is associated to a class 5 day. The model output was calculated at 4.7975, which forecasts a class 5 day.

Once again, we applied our algorithm and evaluated the model output on the two trajectories x and $\tilde{x}$ as seen in figure 6.

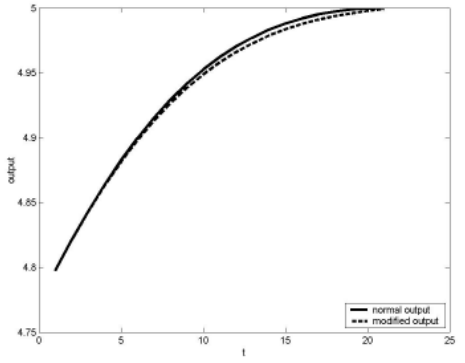

Figure 6. *The model output evaluated on the two trajectories*

The horizontal distance was calculated as 0.1188 and the vertical as 1.0902, giving a total of 1.2089. The second nearest cluster centre was also associated to a class 5 day and produced a calculated total distance of 0.8629. The third nearest cluster centre was associated to the correct class 3 and was the following

$$x_T^T = \begin{bmatrix} 192 & 25 & 34.8 & 19.7 & 37.1 & 19.2 & 1.556 & 0.6875 \end{bmatrix}$$

The model output evaluated on the two trajectories x and $\tilde{x}$ can be seen in figure 7

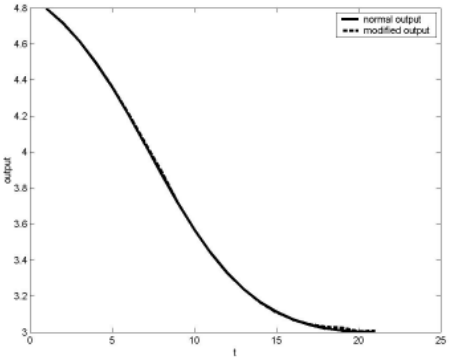

Figure 7. *The model output evaluated on the two trajectories*

When the distances were calculated for these two trajectories we found 0.3988 for the horizontal and 0.0922 for the vertical, making a total of 0.4909.

As for the previous experiment, we see that a cluster centre belonging to the correct class is at a smaller distance than two cluster centres belonging to the wrong class.

6. Conclusions and perspectives

We have proposed a method of measuring distances between data points based on horizontal and vertical vector fields. Our approach is practical and all calculations are carried out locally. Of course, a very large class of nonlinear systems can be analysed by carrying out local calculations and piecing them together. This is what we do.

Our work is ongoing and very much in its infancy. We are conducting trials on the data that we have in order to prepare our model for the Summer season of 2005. We need to run some lengthy computations so that we can associate confidence levels to cluster centres. Then, our proposed distance measure will be applied. We note that Euclidian distances can, in some cases, be misleading.

We believe that the fibred structure approach to analysing nonlinear mappings is very interesting and has a lot of promise. We are continuing to develop computer codes to carry out the necessary calculations.

7. References

Amari S., Nagaoka H., *Methods of Information Geometry*, Oxford University Press, Oxford, 2000.

Chiu S., « Fuzzy Model Identification Based on Cluster Estimation », *Journal of Intelligent and Fuzzy systems*, vol. 2, n° 3, p. 267-278, 1994.

Ehresmann C., « Les connexions infinitésimales dans un espace fibré », *Colloque de Topologie*, Bruxelles, p. 29-55, 1950.

Gerald C., *Applied Numerical Analysis*, Addison-Wesley, 2nd Edition, New York, 1980.

Golub G., Van Loan C., *Matrix Computations*, North Oxford Academic, Oxford, 1983.

Isidori A., *Nonlinear Control Systems*, Springer-Verlag, 2nd Edition, London, 1989.

Mounier G., Couach O., Batton-Hubert M., Clappier O., « Photochemical eulerian modelling using multinesting methodology, application to Rhône-Alpes district », *Proceedings of the 2nd International Conference on Air Pollution Modelling and Simulation*, Champs sur Marne, France, 2001.

Olver P., *Applications of Lie Groups to the solution of differential equations*, Springer Verlag, New York, 1986.

Pearson D., « Approximating Vertical Vector Fields for Feedforward Neural Networks », *Applied Mathematics Letters*, vol. 9, n° 2, p. 61-64, 1996.

Pearson D., Batton-Hubert M., Dray G., « Vertical Vector Fields and Neural Networks: An Application in Atmospheric Pollution Forecasting », *Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms*, Roanne, France, 2003.

Pearson D., Batton-Hubert M., Dray G., « The Use of Vertical Fields and Neural Networks in Atmospheric Pollution Forecasting », *International Journal of Systems Science*, To appear.

Pearson D., Batton-Hubert M., Garcia G., « Predicting Ozone Peaks: A combined CBR and cell mapping approach », *IEMSS02*, Lugano, Switzerland, 2002.

Wonham W., *Linear Multivariable Control: a Geometric Approach*, Springer Verlag, 2nd Edition, New York, 1979.